

JP6202945

Publication Title:

SMALL ADDRESS- BASED MEMORY ACCESS METHOD AND COMPUTER SYSTEM

Abstract:

Abstract of JP 6202945

(A) PURPOSE: To enable a host (hypervisor) to perform access to an arbitrary storage location in an arbitrary guest zone in a large-sized memory when the host and a guest (OS) have small addresses. CONSTITUTION: A specific number of windows used by a host is provided. Each CPU in a system has one or a plurality of window address registers(WAR) 545 and one or a plurality of window registers 546-547. The host designates each page frame(PF) used as a host window in a host zone by using a WAR load instruction and each PF is correlated with the corresponding window number. When the host executes an arbitrary instruction having a small address by accessing the host window PF related to the said WR, the small address of a host operand is automatically replaced with a guest large address in the WR.

Courtesy of <http://v3.espacenet.com>

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開平6-202945

(43)公開日 平成6年(1994)7月22日

(51) Int.Cl.⁵

G 0 6 F 12/06

識別記号

5 6 0 B 9366-5B

F I

技術表示箇所

審査請求 有 請求項の数15 (全 21 頁)

(21)出願番号 特願平5-275633

(22) 出願日 平成5年(1993)11月4日

(31)優先権主張番号 9 7 4 3 9 3

(32)優先日 1992年11月10日

(33) 優先権主張国 米国 (US)

(71)出願人 390009531

インターナショナル・ビジネス・マシーンズ・コーポレーション

INTERNATIONAL BUSIN
ESS MASCHINES CORPO
RATION

アメリカ合衆国10504、ニューヨーク州
アーモンク (番地なし)

(72)発明者 ジョネル・ジョージ

アメリカ合衆国12569、ニューヨーク州プレザント・バレー、プラトーン・ロード、アール・アール 6、ボックス 12

(74)代理人 弁理士 合田 潔 (外3名)

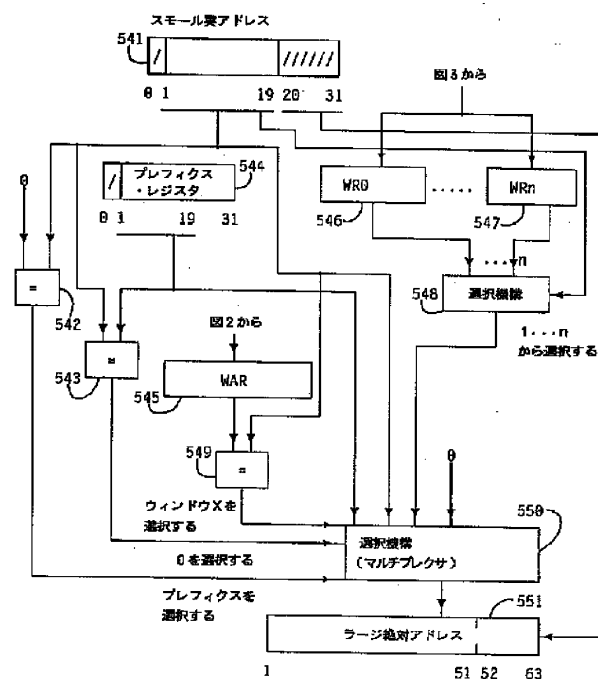
[最終頁に続く](#)

(54) 【発明の名称】 スモール・アドレスを使用するメモリ・アクセス方法およびコンピュータ・システム

(57) 【要約】

【目的】 ホスト（ハイパーバイザ）およびゲスト（OS）がスモール・アドレスを有する場合に、当該ホストが大型メモリ内の任意のゲスト・ゾーンの任意の記憶位置にアクセスできるようにする。

【構成】 ホストが使用する特定数のウィンドウが提供される。システム内の各CPUは、1つまたは複数のウィンドウ・アクセス・レジスタ（WAR）と、1つまたは複数のウィンドウ・レジスタ（WR）を有する。ホストは、WARロード命令を使用して、ホスト・ウィンドウとして使用される各ページ・フレーム（PF）をホスト・ゾーン内で指定し、各PFはそれぞれのウィンドウ番号と関連付けられる。ホストが、当該WRと関連するホスト・ウィンドウPFにアクセスして、スモール・アドレスを有する任意の命令を実行すると、自動的にホスト・オペランドのスモール・アドレスがWR内のゲスト・ラージ・アドレスで置き換えられる。



【特許請求の範囲】

【請求項1】 1つまたは複数のCPUおよび大型メモリを具備し、1つまたは複数のゲスト・システム（ゲスト）およびホスト・システム（ホスト）がそれぞれのローカルなメモリ・ゾーン以外の記憶位置にアクセスできないスモール・アドレスを有する命令オペランドを使用するコンピュータ・システムにおいて、該ホストが、大型メモリ内の任意のゲスト・ゾーンの任意の記憶位置にアクセスするための方法であって、

上記CPUが、1つまたは複数のウィンドウ・アクセス・レジスタ（WAR）と、該CPU用に提供された1つまたは複数の指定ウィンドウにそれぞれが関連する1つまたは複数のウィンドウ・レジスタ（WR）とを含み、ホスト・メモリ・ゾーン内の1つまたは複数のページ・フレーム（PF）を位置指定し、該位置指定された各PFを、CPUの指定ウィンドウと関連付けてホスト・ウィンドウPFにするためのWARロード命令を実行するステップと、

ゲスト識別子およびゲスト・スモール・アドレスによって示されるゲスト・ゾーン内のゲスト位置を、指定ウィンドウおよび該指定ウィンドウに関連付けられたWRと関連付けるためのウィンドウ活動化命令を実行するステップと、

ホスト・オペランドが指定ウィンドウと関連付けられたホスト・ウィンドウPFをアドレスするとき上記ホストがゲスト位置にアクセスするようにする上記ゲスト識別子およびゲスト・スモール・アドレスから大型メモリ用のラージ絶対アドレスを生成し、上記指定ウィンドウと関連付けられたWRに該ラージ絶対アドレスをロードするステップとを含むメモリ・アクセス方法。

【請求項2】 ホスト・ウィンドウPFをアドレス指定するオペランド・スモール・アドレスを有するホスト命令を実行するステップと、

ホストが、上記オペランド・スモール・アドレスによって、ラージ絶対アドレスで位置指定されるゲスト位置にアクセスできるようにするために、CPUのハードウェア／マイクロコードによって、該オペランド・スモール・アドレスを上記WR内のラージ絶対アドレスで置き換えるステップとを含む、請求項1に記載のメモリ・アクセス方法。

【請求項3】 ホスト・ウィンドウPF以外のホスト・ゾーン内のPFをアドレス指定するオペランド・スモール・アドレスを有するホスト命令を実行するステップと、CPUのハードウェア／マイクロコードによって、上記オペランド・スモール・アドレスでアドレス指定される上記ホスト・ゾーン内の記憶位置にアクセスするステップとを含む、請求項1に記載のメモリ・アクセス方法。

【請求項4】 それぞれが、ゾーン識別子用フィールドと、原点ラージ・アドレス用フィールドと、ゲスト・ゾーンの境界を定義するための限界ラージ・アドレス用の

フィールドとを有する複数のエントリを具備し、大型メモリ内の各ゲスト・ゾーンを位置指定するゾーン情報テーブル（ZIT）をメモリ内で提供するステップと、上記限界ラージ・アドレスを、上記生成済みラージ絶対アドレスと比較し、該限界ラージ・アドレスを上回る生成済みラージ・アドレスを拒絶して、ホストが無効なゲスト・アドレスを使用するのを防止するステップとを含む、請求項1に記載のメモリ・アクセス方法。

【請求項5】 それぞれが、ゾーン番号フィールドと、ゲスト・スモール・アドレス・フィールドと、WRに現在関連しているゲスト位置を示すためのラージ絶対アドレス・フィールドとを含むとともに、CPU内の指定ウィンドウおよび関連WRとそれぞれが関連付けられた複数のエントリを具備するCPUウィンドウ指定テーブル（WST）を提供するステップを含む、請求項1に記載のメモリ・アクセス方法。

【請求項6】 上記ウィンドウ活動化命令を実行するとき、指定ウィンドウ用のゲスト・ゾーン識別子およびゲスト・スモール・アドレスを上記WSTのエントリに書き込むステップと、

生成されたゲスト・ラージ絶対アドレスを、上記ゲスト・スモール・アドレスが書き込まれた同じWSTエントリに書き込むステップを含む、請求項5に記載のメモリ・アクセス方法。

【請求項7】 ホスト・ウィンドウPFとして指定されるホスト・ゾーン内の1つまたは複数の連続するPFを位置指定するためのアドレスを、CPUの汎用レジスタにロードするステップと、

汎用レジスタ中のアドレスをWARにロードするWARロード命令を実行して、アドレス指定されたPFを、ホストが使用できるホスト・ウィンドウPFとして指定するステップとを含む、請求項1に記載の、メモリ・アクセス方法。

【請求項8】 ホストによってアクセスされるゲスト・ゾーン内のゲスト位置についての、ウィンドウ識別子、スモール・アドレス、およびゲスト・ゾーン識別子を1つまたは複数のCPU汎用レジスタにロードするステップと、

汎用レジスタの内容を使用してウィンドウ活動化命令を実行し、ラージ絶対アドレスを生成すると共に、該汎用レジスタ中のウィンドウ番号に関連するWRにラージ絶対アドレスをロードするステップを含む、請求項4に記載のメモリ・アクセス方法。

【請求項9】 ゲスト・ゾーン用のZITエントリ中の原点ラージ・アドレスに汎用レジスタ中のゲスト・スモール・アドレスを追加することによって、ラージ絶対アドレスを生成するステップを含む、請求項8に記載のメモリ・アクセス方法。

【請求項10】 ゲストがゲスト命令を実行している間にゲスト動作をインタセプトするステップと、

ホスト・テーブル中のゲスト状態記述子エントリに、インタセプトされたゲスト命令の記憶位置であるスモール・アドレスを取り込み、当該エントリのゲスト・ゾーン識別子を示すステップと、

ホストの指定ウィンドウ用のウィンドウ活動化命令を実行して、ゲスト・ゾーン中のインタセプトされたスモール・アドレス用のラージ・アドレスを生成し、該指定ウィンドウに関連するWRにそのラージ・アドレスを格納するステップと、

指定ウィンドウと関連するホスト・ウィンドウPFをアドレス指定するホスト・オペランド・アドレスを有するホスト命令を実行するとき、ホストがゲスト命令にアクセスして、ゲスト・ゾーンにゲスト命令を取り込むために、ホスト・オペランド・アドレスをWR中のラージ・アドレスで置き換えるステップを含む、

請求項1に記載のメモリ・アクセス方法。

【請求項11】ゲスト・スモール・アドレスを、プレフィクス・レジスタ中のプレフィクス・アドレスおよびアドレス・ゼロと同時に比較して、ゲスト・スモール・アドレスがプレフィクス・アドレスであるか、または逆プレフィクス・アドレスであるかを判定するステップと、プレフィクス・レジスタまたはアドレス・ゼロに関してcompare equal条件が得られた場合に、WRからラージ・アドレスを出力する代りに、プレフィクス・レジスタからプレフィクス・アドレスを出力し、またはアドレス・ゼロを逆プレフィクス・ラージ・アドレスとして出力するステップを含む、請求項1または10に記載の、ホストが、大型メモリ内の任意のゲスト・ゾーンの任意の位置にアクセスできるようにする方法。

【請求項12】単一のWARを使用して、ホストに対して指定されたウィンドウと関連する単一のホスト・ウィンドウPFを位置指定するステップ、または複数のWARを使用して、それぞれのWARが、ホストに対して指定されたそれぞれのウィンドウと関連する単一のホスト・ウィンドウPFを位置指定するステップ、または単一のWARを使用して、それぞれがホストに対して指定されたウィンドウと関連する複数のホスト・ウィンドウPFを位置指定するステップのいずれか1を含む、請求項1に記載のメモリ・アクセス方法。

【請求項13】プログラムによってアクセス可能でないシステム・エリア記憶域内の活動ZIT中の対応するエントリと異なるエントリを有する置換ZITを、大型メモリ内のプログラム・アクセス可能エリアに書き込み、活動ZITだけでシステムを制御するステップと、システム内のすべてのCPUによる実行を中断するステップと、

すべてのCPUの実行が中断されている間に、システム内の選択されたプロセッサによって特殊コマンドを発行して、活動ZITを置換ZITでオーバーレイして、置換ZITを活動化するステップと、

各エントリに含まれるゲスト・スモール・アドレスとゾーン識別子を使用し、新たに活動化されたZIT中の識別ゾーン用のエントリを使用することによって、CPUウィンドウ指定テーブル中の各エントリに含まれるラージ・アドレスを再計算し、CPUウィンドウ指定テーブル・エントリおよび関連WRに、再計算されたラージ・アドレスを再ロードするステップと、

CPUが新たに活動化されたZIT中でゾーンを使用できるように、すべてのCPUの命令実行を再開して、システム内でのゾーンの再構成を完了するステップとを含む、システムを動的に再構成できる際に使用する、請求項4に記載のメモリ・アクセス方法。

【請求項14】1つまたは複数のCPUおよび大型メモリを具備し、1つまたは複数のゲスト・システム（ゲスト）およびホスト・システム（ホスト）が、メモリ内のそれぞれのローカルなメモリ・ゾーン以外の記憶位置にはアクセスできないスモール・アドレスを命令オペランドとして含むコンピュータ・システムにおいて、ホスト・メモリ・ゾーン内の1つまたは複数のページ・フレーム（PF）を位置指定するためのホスト・スモール・アドレスを含む、1つまたは複数のウィンドウ・アクセス・レジスタ（WAR）と、

それぞれが、1つまたは複数のホスト・ウィンドウに関連付けられ、ゲスト位置をアクセスするためのラージ絶対アドレスを含む、1つまたは複数のウィンドウ・レジスタ（WR）と、

WARロード命令に応答して、上記PFを上記ウィンドウと関連付ける手段と、

ウィンドウ活動化命令に応答して、ゲスト識別子およびゲスト・スモール・アドレスによって示されるゲスト・ゾーン内のゲスト位置を、上記ウィンドウおよびWRと関連付ける手段と、

上記ゲスト識別子およびゲスト・スモール・アドレスの内容に基づいて、上記ラージ絶対アドレスを生成し、該ラージ絶対アドレスを上記WRにロードする手段とを有する、コンピュータ・システム。

【請求項15】それぞれが、ゾーン識別子用フィールドと、原点ラージ・アドレス用フィールドと、ゲスト・ゾーンの境界を定義するための限界ラージ・アドレス用フィールドとを有する複数のエントリを具備し、大型メモリ内の各ゲスト・ゾーンを位置指定するゾーン情報テーブル（ZIT）を有する、請求項14に記載のコンピュータ・システム。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は、ホストとそのゲストが、それぞれのローカル・ゾーン内の記憶位置にしかアドレスできないスモール・アドレスを使用中のときに、ホストが大型メモリにある任意のゲスト・ゾーンの任意の記憶位置にアクセスできるようにするものである。

【0002】

【従来の技術】本明細書で「スモール・アドレス（小型もしくは小規模アドレスともいう）」とは、16ビット、20ビット、24ビット、31ビット、または32ビットのいずれかの一般アドレス・サイズを使用するものと定義する。本明細書で「ラージ・アドレス（大型もしくは大規模アドレスともいう）」とは、使用中のスモール・アドレスよりも大きなサイズを持つものと定義する。多数の既存のプログラムがスモール・サイズの仮想アドレスを使用しており、それらのアドレスは、スモール実アドレスまたはスモール絶対アドレスに変換される。変換された実アドレスまたは絶対アドレスが、メモリにアクセスするのに使用される。

【0003】従来の設計のホスト（ハイパバイザ）においては、ホストが使用するサイズよりも大きなアドレス・サイズを必要とする大型メモリ内のどの位置にもアクセスすることができなかった。従来型のホストおよびそのゲスト（オペレーティング・システム、OS）は、スモール・サイズ・アドレス（たとえば、最大 2^{31} バイト・サイズ・メモリしかアドレスできない31ビット・アドレス）を使用するように設計されていた。OSはそれぞれ、コンピュータ・システムのメモリ内の異なるゾーンに割り当てられた。各ゾーンのサイズは、メモリ・サイズの一部分であった。ホスト（ハイパバイザ）は、OSを監視したが、あらゆるゲストのゾーン内の任意の位置にアドレスできなければならなかった。

【0004】スモール・アドレスは、米国特許第4843541号に開示された従来の論理的に区分されたシステムで使用されている。このシステムでは、各ゲスト（guest）が異なるメモリ・ゾーンに配置され、かつ各ゲストが、IBM MVS（多重仮想記憶域）オペレーティング・システムやVM（仮想計算機）オペレーティング・システムのコピーなどのオペレーティング・システム（OS）となっている。ホスト（host）は、システムの異なる区画にあるすべてのゲスト・オペレーティング・システムを監視するシステム、たとえば（VMのような）ハイパバイザ・プログラムである。ゾーン（zone）は、システム主記憶装置内の別個の連続した区域であり、システムのすべての資源から構成された異なる論理区画に割り当てられる。このようなシステムを中央電子複合体（CEC）と呼ぶ。ゲストおよびホストはそれぞれ、31ビットのスモール・アドレス・サイズを使用する。論理的に区分されたシステムを制御するためのすべての既知のハイパバイザは、スモール・アドレスしか処理できない。そのようなシステムで使用されている「ウィンドウ」は知られていない。

【0005】「ウィンドウ（window）」は従来、パーソナル・コンピュータ（PC）で使用されており、PC DOSソフトウェア・オペレーティング・システムが使用する最下位1MBの記憶域における4つの未使用16

KB区域に配置される。ウィンドウを使用して、最下位1MBのメモリと、1MBを上回る最大拡張アドレス $2^{24}-1$ までの拡張メモリ・アドレスの間で、16KBページ分のデータが転送されていた。そのようなウィンドウによって、PCは最大16MBのメモリを使用することができた。そのような従来技術の例には、米国特許第4943910号がある。

【0006】大型メモリ・アドレスは、“Guest/Host Large Addressing Method and Means”と題する米国特許出願第816911号（特願平4-329167号）に開示されている。この出願は、ゲストのスモール・アドレスまたはラージ・アドレスをラージ絶対アドレスに変換する方法を提供している。これは、アドレス指定にウィンドウを使用していない。

【0007】ラージ・アドレスを扱った別の特許出願には、“Large Logical Addressing Method and Means”と題する米国特許出願第803320号（特願平4-288025号）がある。この出願は、ゲスト・スモール・アドレスを使用して、大型メモリをアドレスするためのラージ・アドレスを生成している。これは、アドレス指定にウィンドウを使用していない。

【0008】

【発明が解決しようとする課題】メインフレームおよびその他のコンピュータのシステム・メモリのサイズを大幅に拡張すると、それらのコンピュータは、従来よりはるかに多くのデータおよびプログラムを格納し、従来使用していたよりも多くの機能を実行することができる。たとえば、メインフレーム内により多くの論理的区画を設けて、大型メモリにおけるゾーンの数を増やし、サポートできるゲスト・オペレーティング・システムの数を増やすことができる。

【0009】ゲストはホスト（ハイパバイザ）によって監視される。ホストは、（VMなどの）ソフトウェア、または（IBM PR/SM（プロセッサ資源/システム管理機能）などの）マイクロコードもしくは内部コードに存在することができる。ホストは、各ゲスト・ゾーン内の記憶位置にアクセスすることができなければならない。

【0010】現在のIBMメインフレームは、複数のゲストと1つのホスト（ハイパバイザ）との間で区分される小型のメイン・メモリ（最大 2^{31} バイト＝2GB）を有する。現行のIBM PR/SMシステムおよびVMシステムでは、ホストは、必要に応じて、31ビットのスモール・アドレスを使用して任意のゲスト位置にアクセスする。このホスト要件により、メモリ・サイズが 2^{31} バイト以下に制限される。なぜなら、ホストは任意のゲスト位置にアクセスできなければならないからである。ゲストも31ビット・アドレスを使用する。

【0011】現在使用している（スモール・サイズ・アドレスを使用するように設計された）ホストおよびゲスト

トを変更してスモール・サイズ・アドレスを超えるラージ・サイズ・アドレスを使用できるようにするのは、コストがかかり面倒である。

【0012】本発明の主目的は、システムのハードウェア／マイクロコードの変更を最小限に抑えてホストと大型メモリを使用する任意のゲストとの間でデータ転送を実行するのに必要なラージ・アドレスを従来型のホストが認識することなく、（スモール・サイズ・アドレスしか使用できない）当該ホストを大型メモリにおいてゲストと共に動作できるようにすることである。

【0013】ゲストの識別信号をゲストのスモール・サイズ・アドレスと組み合わせてホストに与えることにより、ゲストの記憶位置をホストに示すことができる。

【0014】したがって、本発明の別の目的は、ホストが従来型のスモール・アドレス（たとえば、31ビット）を使用して、（たとえ、ゲストの記憶位置が、ホストが使用中のスモール・アドレスの範囲外である場合でも）大型メモリ内の任意のゲストの記憶位置にアクセスできるようにすることである。

【0015】

【課題を解決するための手段】本発明によれば、ホストが、スモール・アドレスを使用して大型メモリ内の任意のゲストの記憶位置（ゲスト位置）にアクセスする能力は、次のようにして獲得される。1. ホスト・ゾーン内にウィンドウ・ページ・フレーム（PF）を定義する。2. ホストPFを、ホストに対して透過的なラージ・ゲスト・アドレスと関連付ける。ホストは、ゲスト位置をアドレスしたい場合、スモール・アドレスを使用してウィンドウPFにアドレスする。そうすると、CPUのハードウェア／マイクロコード（または内部コード）が、そのホスト・スモール・アドレスを、関連するラージ・アドレスで置き換え、その後は、ラージ・アドレスが、任意のゲスト・ゾーンにおける必要なゲスト位置を指定する。

【0016】任意のゲスト位置に対する実ゲスト・アドレス（ラージ・アドレス）は、従来型のスモール・アドレスしか処理しないホスト・プログラムには見えない。

（ホストおよびゲストに対して透過的な）CPUハードウェア／マイクロコードは自動的に、任意のゲスト位置がいつホスト・アテンションを受け取るかを検出し、ゲスト・ゾーンおよびそのゲスト位置に対するゲスト・スモール・アドレスをホストに対して識別する。この情報により、そのゲスト・ゾーンおよびスモール・アドレスを透過的に使用して、ホスト・プログラムが（ホスト・スモール・アドレスによって位置指定される）ホスト・ウィンドウを介して間接的に使用できるラージ絶対アドレスをセットアップするハードウェア／マイクロコードをホストが呼び出すことができる。

【0017】したがって、ホスト・プログラムおよびゲスト・プログラムはそれぞれ、従来型のスモール・アド

レスを使用しているものと仮定する。CPUハードウェア／マイクロコードは、スモール・アドレスの値に応じて、ゲストのゾーン原点にスモール・アドレスを加え、あるいはスモール・アドレスにプレフィクシング（prefixing）し、あるいはスモール・アドレスに逆プレフィクシング（reverse prefixing）することによって、ゲスト・スモール・アドレスをラージ・アドレスに変更する。

【0018】従来型のホスト・プログラムがシステム上で本発明を使用するには、ホスト・プログラムを、ゲスト位置ではなくホスト・ウィンドウPFをアドレスできるように変更するだけでよい。CPUハードウェアおよびマイクロコードは、ホスト命令がスモール・アドレスを使用してゲスト位置をアドレスするとき、ホストがウィンドウPFを定義し、隠れたウィンドウ・レジスタ（WR）のゲスト・ラージ・アドレスを提供できるようにするだけでよい。

【0019】本発明は次のように実施される。1. 1つまたは複数のホスト・ウィンドウPFを位置指定するホスト・スモール・アドレスを含むウィンドウ・アクセス・レジスタ（WAR）によって、ホスト・ゾーン内で各ホスト・ウィンドウPFを位置指定即ち特定する。2. 各ホスト・ウィンドウPFを、ラージ絶対アドレスを収容するウィンドウ・レジスタ（WR）と関連付ける。

【0020】WARは、次の2つの代替方法のどちらかで実施される。1. 1つまたは複数のWARを提供し、各WARが単一のホストPFを位置指定する。2. 単一のWARを提供し、このWARの内容が複数の連続するホストPFを位置指定する。これらの複数のPFの中から、（複数の連続するホスト・ウィンドウPFの組における）特定のホスト・ウィンドウPFが、ホスト・ウィンドウPFにアクセスするために提供された各ホスト・スモール・アドレスにおけるPFアドレスの下位ビットによって選択される。複数の異なるホスト・ウィンドウPFのアドレスは、単一のWARで定義されるときでも、ホストが独立に使用することができる。すなわち、ホストは、同一のゲスト・ゾーンまたは複数のゲスト・ゾーンにおける異なるゲスト位置にアクセスするために、複数のウィンドウPFを同時に使用することができる。

【0021】各WARの内容は、ホストによって指定されるホスト・ゾーンのウィンドウPFを位置指定するスモール・アドレスをWARにロードする命令を実行するホストによって構成される。

【0022】1つまたは複数のWRの存在、並びにWRとホスト・ウィンドウ（WRと同数）との関連付けは、システムのハードウェア／マイクロコードの構造によって決まる。ただし、WRとホスト・ゾーンPFとの関連付けは、「WARロード（LOAD WAR）」命令が実行されるまで決定されず、WRと特定のゲスト・スモール・ア

ドレスとの関連付けは、「ウィンドウ活動化(Activate Window)」命令が実行されるまで決定されない。

【0023】さらに一般的に言うと、単一のWARによってゾーン内で位置指定される連続するウィンドウPFの数は、 2^P (Pは、ウィンドウにアクセスするためのゾーン・オペランド・アドレスにおけるPFコンポーネントの最下位ビットの数を表す) によって定義される。ここで、したがって、ゾーン・オペランド・アドレスは、命令によってホスト・ゾーン内に提供され、そのようなアドレスは、指定されたホスト・ウィンドウPFセ

ットにおける必要なウィンドウPFを自動的に選択する。これは、ゾーン・オペランド・アドレスが、ウィンドウPFの選択を自動的に実行するP個の最下位ビットを持つ「PFアドレス・コンポーネント」を含むからである。

【0024】ゾーン内の各ウィンドウは、ウィンドウ・レジスタ(WR) およびウィンドウ空間と関連付けられる。ウィンドウ空間とウィンドウの関連付けは、ホスト・ゾーンで実行されるWARロード命令によって明示的に実行される。WRとウィンドウの関連付けは、本発明

を提供するCPUハードウェアまたはマイクロコードの設計中で暗示的に実行される。

【0025】実行するホスト命令オペランドが任意のホスト・ウィンドウPFアドレスに等しいとき、CPU内のハードウェア/マイクロコードが、そのホスト・ウィンドウPFに関連付けられたWRに含まれるラージ絶対アドレスを自動的に置き換える。この動作によって、関連付けられたWR中にラージ・アドレスを持つゲスト位置に対する、ホスト・オペランドによる自動的な間接ラージ・アドレス指定が実行される。

【0026】ホスト命令オペランド・アドレスがどのホスト・ウィンドウPFアドレスとも等しくないときは、ホスト・ウィンドウは選択されず、間接ラージ・アドレス指定動作は行われぬ。この場合、ホスト命令オペランド・アドレスは、ホスト・ゾーンにおいて、従来技術の場合と同様に、プレフィクシングおよび逆プレフィクシングを含めて、通常のホスト・アドレスとしてアクセスされる。

【0027】記憶域オペランドを持つどんな種類のホスト命令も、ホスト・ウィンドウ・アドレスを指定して、関連するゲスト・アドレスにアクセスすることができる。どんな種類のホスト命令動作もゲスト・ゾーンで行うことができる。たとえば、ホストは、それぞれ2つの異なるWRが関連付けられている2つの異なるゲストの2つのオペランド用に2つの異なるホスト・ウィンドウPFを指定することによって、2つの異なるゲスト位置の間で記憶域オペランドを移動することができる。

【0028】したがって、本発明の他の目的は、ホストがスモール・アドレスを使用して、2つの異なるゲスト位置の間でデータを移動/コピーし、あるいは同一のゲ

スト・ゾーン内の2つの記憶位置の間でデータを移動/コピーできるようにすることである。

【0029】ゾーン情報テーブル(ZIT)は、システムに常駐するそれぞれのゲストのゾーンを定義する。ハードウェア/マイクロコードでサポートされるWR活動化プロセスは、要求側ゲストのスモール・アドレスを、それぞれのゲストのゾーン用のZITエントリ中のゾーン限界情報と突き合わせて検査して、スモール・アドレスがゲストのゾーンのアドレス指定範囲内にあることを確認し、その後、ラージ・アドレスをWRに書き込む。

【0030】WRに(ゲストのスモール・アドレスを表す)ラージ・アドレスをロードするためのプロセスは、システム内の仮想マシンのインタセプト事象または割込み事象の検出によって呼び出すことができる。そのような事象の存在は、SIE(解釈実行開始)命令の状態記述(SD)に記憶された信号によって示すことができる。システム内の各ゲストにはそれぞれSDが割り当てられる。SDとは、システムが各ゲストの動作を処理するのに必要とするすべての情報を記憶する主記憶装置内の制御ブロックである。ホストに利用可能なSD内のインタセプト、割込み、およびそれらのスモール・アドレスを示す方法は、"IBM System/370 Extended Architecture Interpretive Execution"と題する1985年のIBM刊行物SA22-7095-1に詳細に記載されている。

【0031】本発明で利用できる、任意のSD内の重要な情報としては、インタセプト事象および割込み事象の発生を示す標識、および現インタセプト事象または割込み事象の影響を受ける、ゲスト・ゾーン内の記憶位置を指定するスモール・アドレスがある。

【0032】ホストは、ゲストのインタセプト指示または割込み指示に応じてゲストのSDからホストに渡される情報に基づき、ゲスト・ゾーン内の、ホストによってアクセスされる各ゲスト・オペランドにホスト・ウィンドウを割り当てる。

【0033】本発明のもう1つの特徴は、システム動作中の任意の時にZIT(ゾーン情報テーブル)を再構成することによって、システム内のホストおよびゲスト用のゾーンを動的に再定義できることである。

【0034】ホストは、任意のゲストのゾーンにアクセスする固有の権限を持つ。この権限は、ホストが、ゲストの処理によって誤って発生する場合がある予測できない問題からゲストを保護するのに肝要である。

【0035】ホスト・ゾーンが、大型メモリ内の(最下位アドレス範囲を持つ)メモリの第1ゾーンである必要はない。ホスト・ゾーンは、大型メモリに適用されるどのスモール・アドレス指定機能の範囲外であってもよい(これは、PC拡張メモリをサポートするために使用される従来技術のPCウィンドウとは異なる)。

【0036】本発明により、コンピュータ・システム内でハードウェアまたはマイクロコードを単純に追加することによって、スモール・アドレスを使用するホスト・プログラムが、大型メモリ内の任意の位置にアクセスすることが可能となる。

【0037】

【実施例】

大型メモリのマップー図1

本発明の好ましい実施例では、メインフレームの大型メモリ内に1つのホスト（ハイパバイザ）と複数のゲスト（オペレーティング・システム）が配置され、メモリ・サイズは2ギガバイト（2GB）を上回る実記憶域であり（2GBの数倍であることが好ましい）、ホストは、2GBの範囲内しかアドレスできない31ビットのスモール・アドレス（small address）を使用する。

【0038】図1は、大型メモリ内の領域を1つのホスト、複数のゲスト、および「システム・エリア記憶域」（すべて、システム主記憶域ハードウェア・エンティティの一部）に割り当てたマップを示す。しかし、システム・エリア記憶域にアドレス指定可能なのはハードウェアおよびマイクロコード（内部コード）だけであり、システム内のCPU上で走行するプログラムによってアドレスすることはできない。ホスト・ゾーンおよびゲスト・ゾーンはそれぞれ、メモリ・エンティティのプログラム・アドレス指定可能部分に配置されている。各ゾーンのサイズは、 2^{31} バイト未満である。

【0039】各ゾーン（ホストまたはゲスト）は、メモリ・エンティティ内で空間が存在する任意の場所に配置することができ、かつメモリ・エンティティ内に任意の順序で配置することができる。各ゾーン内のメモリ空間は連続している。

【0040】ホストおよび各ゲストは、スモール・アドレスを使用する。好ましい実施例では、スモール・アドレスは31ビットのサイズを有する。スモール・アドレスは、単一のゾーン内の記憶位置にしかアドレスできない。これは、各ゲストによって発行される各メモリ・アドレスに対する限界検査によって実施される。

【0041】実メモリ空間は、「ページ・フレーム（PF）」と称する単位に分けられる。ページ・フレームは、アドレス変換および再配置に使用される。好ましい実施例では、実メモリ空間が4キロバイト（KB）のページ・フレーム（PF）に分けられ、メモリ内の各PFが4KB境界上に配置される。

【0042】従来の方法では、まずゲストの各仮想アドレスが実アドレスに変換される。次に、実アドレスが、IBM ESA/390システム用の従来技術による方法でアドレス・プレフィクシングを使用することによって絶対アドレスに変換される。この場合、ゼロの実アドレスと、CPUのプレフィクス・アドレスに等しい実アドレスとを除き、すべての実アドレスは、それに対応す

る絶対アドレスと同じである。

【0043】本明細書では、すべてのアドレスは、メモリ・アクセスに使用する際は絶対アドレスである。

【0044】好ましい実施例では、ホスト・ゾーン内に2つのウィンドウを提供し、これらのウィンドウを使用して、ラージ・アドレス（large address）を必要とする大型実メモリ内における、他の任意の1つのゾーン内もしくは他の任意の2つのゾーン内の任意の2つの記憶位置にアクセスする。

【0045】ホストは、そのゾーン内の任意の場所にこれらの2つのウィンドウを配置することができる。ウィンドウは、ホスト・ゾーン内のデータにアクセスするホスト命令オペランドに妨害を与える可能性が最も少ない場所に配置することが好ましい。

【0046】各ウィンドウには、ホスト・ゾーン内のページ・フレーム（PF）が割り当てられる。好ましい実施例では、1つのウィンドウ・アクセス・レジスタ（WAR）によって2つのウィンドウを位置指定する（locate）。この割当ては、図2に示す「WARロード（load WAR）」命令を実行することによって行う。

【0047】しかし、図3には「WARロード」命令の別の実施例が示されている。この場合、複数のWARが、ホスト・ゾーンにおけるそれぞれのウィンドウとして、不連続のPFを位置指定する。この実施例では、図3に示す「WARロード」命令を使用するが、この命令は複数のWARのうちの1つに単一のホストPFを割り当てる。したがって、複数のWARの各々にホストPFを割り当てるために、WARの数と同じ数のWARロード命令を実行する必要がある。

【0048】WR自体は、ホストまたは任意のゲストにとって明らかである必要はない。WRの存在を、ホストまたはゲストのいずれかのプログラム・コードで示す必要はない。

【0049】ここに示す好ましい実施例では、ホスト・ゾーンだけがウィンドウを使用し、ゲスト・ゾーンはウィンドウをもたないものと仮定する。WARは、システムの初期設定時にロードされる。

【0050】単一のWARが2つのウィンドウPFを位置指定する好ましい実施例では、ホスト・オペランドのPFアドレス・コンポーネントの最下位ビットの状態が、これらの2つのウィンドウPFのうちどちらをアドレスするかを指定する（4つのホスト・ウィンドウPFが提供されている場合は、オペランド・アドレスPFコンポーネントの最下位2ビット（31ビット・アドレスのビット18およびビット19）の状態を使用して、アドレスされるウィンドウPFを選択する）。

【0051】さらに詳細に述べると、ホストに2つのウィンドウ番号が提供され、各ウィンドウ番号は、暗示的に関連付けられたWRを持つ。WRは、マシンの設計時に、またはマシン・マイクロコードのあるバージョンが

マシンにロードされるときに固定される。好ましい実施例で指定されるWR番号はウィンドウ番号に順次固定され、ウィンドウ1にはWR0が、ウィンドウ2にはWR1が割り当てられる。ウィンドウ番号がこれより多い場合、それらの番号はそれぞれ異なるWRと関連付けられる。したがって、各ウィンドウ番号は、暗示的に関連付けられたWRを持つ。

【0052】好ましい実施例ではまた、WRはそれぞれ、大型メモリ内の任意の位置にアクセスするために63ビット・アドレスを含むことができる。

【0053】ウィンドウ数が固定され、WRとホスト・ゾーンが利用できるウィンドウとの関連付けが固定されているので、ホストはWARロード命令を使用して各ウィンドウ番号にウィンドウ空間を割り当てる必要がある。

【0054】ホスト・ゾーンにおけるウィンドウの位置指定—図2

各ホスト・ウィンドウに対するホストPFの割当ては、ホストで実行される「WARロード」命令によって行う。このホストによるPFおよびウィンドウ間の割当てでは、PFとウィンドウを関連付けるプロセスで、WRが、指定されたPFに暗示的に割り当てられる。WARロード命令の実行は、ホストが実行中のシステム内の任意のCPUのハードウェア/マイクロコードによって行われる。

【0055】図2は、WARロード命令用のハードウェアおよびプロセスを示す。この命令は、命令コードを持ち、汎用レジスタR1を指定する単一のオペランドを持つ。命令実行前に、R1のビット位置1ないし18にホストPFアドレスがロードされている。このアドレスは、2つの連続するPFを含む8KBのエリアを位置指定する。割当てプロセスにより、R1のビット1ないし19のPFアドレスが、WARのビット位置1ないし19に移され、WARのビット位置20ないし31にはゼロが書き込まれる。WR0およびWR1は2つのウィンドウと暗示的に関連付けられる。WARでアドレスされるPFはWR0と関連付けられ、もう一方のPFはWR1と関連付けられる。オペランド・アドレスのビット19の状態によって、適切なウィンドウPFが選択される。

【0056】ホストは、各CPU上で、システム内の他のCPUとは独立して「WARロード」命令を実行するので、各CPUにはそのWARがロードされる。希望するなら、異なるCPU内で、複数のホストPFを同一のウィンドウ番号には、割り当てることができる。本明細書での議論では、異なる割当てによって生じる混乱を最小限に抑えるために、すべてのCPUにおいて同じウィンドウ番号には、同一のホストPFアドレスが割り当てられるものと仮定する。

【0057】ホストがゲスト位置にアクセスできるよう

にするには、WRにゲスト・ラージ・アドレスをロードしておく必要がある。任意のゲスト位置にアクセスするために、ホストはどちらかのホスト・ウィンドウを使用することができる。WRにゲスト・ラージ・アドレスをロードした後、ホスト・ソフトウェアは、WARによって現在指定されている任意のウィンドウPF内のバイト位置に対するオペランド・アドレスを持つ任意の命令を実行することができる。その後、ハードウェアは、そのホスト命令を実行する際にホストのスモール・アドレスをラージ・アドレスで置き換えることによって、関連するWR内のラージ・アドレスに暗示的にアクセスする。

【0058】アクセスされるゲスト位置は、どのゲスト・ゾーンのどのバイト境界上にあってもよい。各ホスト・ウィンドウは、PF境界上に配置される。これは、任意のホスト・アドレス用のすべての可能なバイト変位(D)値をウィンドウに収容するために、メモリ空間のPFが必要だからである。

【0059】ウィンドウに対する各ホスト・アドレスはアドレス置換動作を引き起こすので、ホストは、「ウィンドウ活動化」命令でウィンドウが活動化されている間に、任意のホスト・ウィンドウPF宛のどのアドレスを使用しても、任意のホスト・ウィンドウに記憶されたデータを取り出し、あるいはそこに記憶することはできない。したがって、ホストは、ウィンドウのアドレスを使用してアクセスされると予想されるそのどのウィンドウPFにもデータを記憶すべきではない。

【0060】しかし、ホストは、そのデータを収容するための1つまたは複数のウィンドウPFを含む、複数の連続するPFを必要とするラージ・オペランド用に非ウィンドウ・アドレスを使用する場合、任意のウィンドウPFにデータを記憶し、そこから取り出すことができる。たとえば、move-character-long命令中のオペランドは、1つまたは複数のウィンドウPFに及ぶオペランド・データ用の非ウィンドウ・オペランド・アドレスを指定することができる。

【0061】ホストによるゲスト・アクセス用のウィンドウ割当て—図3

ホストは、「ウィンドウ活動化」命令を実行することによって、ゲスト位置にウィンドウ番号を割り当てる。この命令は、ゲスト・ゾーン(R3レジスタ中でホストによって指定される)内のゲスト・スモール・アドレス(R1レジスタ中でホストによって指定される)にウィンドウ番号(R3レジスタのフィールド中でホストによって指定される)を割り当てる。次に、指定されたウィンドウ番号と関連するWRが、その指定されたゲスト位置と暗示的に関連付けられる。この命令は、対応するゲスト・ラージ絶対アドレスを暗示的に算出し、そのアドレスを、関連するWRと、「ウィンドウ活動化」命令を実行するCPUのCPUウィンドウ指定テーブル内のそのWR用のエントリとに入力する。

【0062】ホストは、WRの存在を知る必要がなく、また、ウィンドウを使用する際にWRのラージ・アドレス内容にアクセスできる必要もない。ホストが知る必要があるのは、そのウィンドウ番号だけである。ホスト・ソフトウェアに対して透過的な自動ハードウェア／マイクロコード動作によって、ラージ・アドレスが生成され、ホストが「ウィンドウ活動化」命令を実行するとき、WRにこのラージ・アドレスが入れられ、ホストがホスト・ウィンドウを参照する命令を実行するとき、自動的に、ホスト・オペランドにおけるスモール・ウィンドウ・アドレスがラージ・アドレスで置き換えられる。

【0063】ハードウェア／マイクロコードを使用してゲストのラージ・アドレスを生成しWRに入れるために、ホストが「ウィンドウ活動化」命令を呼び出すことができるようにするには、ホストによる準備動作が必要である。この準備動作には、SDテーブル（図9に表す）にゲストのインタセプト事象および割り込み事象があるか否かをホストに監視させることが含まれる。ホストによる監視は、CPUの信号によって条件付けることができる。この信号は、インタセプト事象または割り込み事象が発生したとき、または、ゲストのいずれかのSDエントリ中におけるインタセプト指示または割り込み指示を検出するためにSDテーブルを監視するタイマ割り込みによって、ホストがそれ自体に定期的に割り込んだときに、レジスタR1およびR3をロードするためにホストの監視動作を呼び出す。

【0064】ホストは、テーブルのゲストSD中でそのような指示を見つけると、図3に示す「ウィンドウ活動化」命令プロセスを実行する前に、そのSDから、R1汎用レジスタ42およびR3汎用レジスタ43の内容を

セットアップするための情報を読み取ることができる。【0065】R1をセットアップするには、ホストがゲスト・アクセスにそのどちらのウィンドウを使用するかを決定し、そのウィンドウ番号をR1に入れる。ゲスト・ゾーン番号（ZN）は、インタセプトされたSDエントリに対するインデックスとして、あるいはそのエントリ内のZNフィールドを読み取ることによって、当該SDエントリによって取り出される。このZN値は、ホストによってR1汎用レジスタ42のZNフィールドにロードされる。

【0066】R3をセットアップするには、アクセスされるゲストのアドレスが、（インタセプト事象または割り込み事象の発生時にマイクロコードによってSDに入れられた）インタセプトされたSDの別のフィールドから取り出され、そのアドレスがホストによってR3汎用レジスタ43にコピーされる。

【0067】その後、図3に表したプロセスを使用してウィンドウ活動化命令を実行することができる。この命令は、R1にある31ビットのゲスト・スモール・アドレスから63ビットのゲスト・ラージ・アドレスを生成

し、その63ビット・アドレスを、ホストで選択されたWRにロードし、またその63ビット・アドレスを、CPUウィンドウ指定テーブル・エントリ47内でそのWRに割り当てられたエントリの「ラージ絶対アドレス」フィールドにロードする。各CPUに「CPUウィンドウ指定テーブル」があり、このテーブルは、プログラムがアドレス指定可能でない「システム・エリア記憶域（図1）」に存在することができる。このCPUテーブルは、それぞれのCPU内の各WRの内容を表すエントリを持つ。WRにロードされる63ビット・アドレスは、ゲスト・ゾーン内のバイト・アドレスである。図8は、関連するCPU内の各WR用のそれぞれのWRエントリを含む、各CPUウィンドウ指定テーブルを表している。各テーブル・エントリは、ゲストのゾーン番号と、ゲストのスモール・アドレスと、CPUテーブル・エントリ中のスモール・ゲスト・アドレスを表すゲストのラージ絶対アドレスとを含む、WRへの現ゲスト・アドレス割当てを含んでいる。

【0068】R1中のZN値は、アクセスされるゲストのゾーンと関連するZITエントリにアクセスするためにZIT44に索引付けするのに使用される。加算機構49は、64ビット加算機構であり、R3レジスタ43中のスモール・ゲスト・アドレスに、ZITエントリから得られたゾーン原点を加算する。この結果、63ビットの絶対アドレスが生成され、このアドレスが比較機構45に提供される。比較機構45は、生成された絶対アドレスを、ZITエントリから得られたゾーン限界と比較する。絶対アドレスがZITエントリのゾーン限界より小さい場合（通常の場合）、R1汎用レジスタ42が供給するWR番号で位置指定されたCPUウィンドウ指定テーブル・エントリ47にこの絶対アドレスが格納される。絶対アドレスがゾーン限界より大きい場合、レジスタ46によってCPUウィンドウ指定テーブル・エントリ47に例外スモール・アドレス値が出力され、例外信号が生成されてCPUに割り込む。CPUウィンドウ指定テーブル・エントリ47にロードされた絶対アドレス値は、次にウィンドウ・レジスタ48にロードされる。ウィンドウ・レジスタ48はこのときラージ・アドレスを含んでおり、関連するウィンドウPFがホスト命令によってアドレスされるとき、それがゲスト・スモール・アドレスと置き換わる。

【0069】CPUウィンドウ指定テーブル・エントリ47は、図7に関連して説明する、ZIT内でのゾーン再配置中に使用される。ゾーン再配置は、特殊コマンドによって開始される。

【0070】ZITを図1に示す。各ゾーンはZITエントリを有し、ZITエントリは、ゾーン番号、ゾーン原点、およびゾーン限界を含むフィールドを有する。エントリ内のゾーン番号を使用して、ZITのアクセスに使用されるR1のZN値を検証することができる。

【0071】WARレジスタおよびWRレジスタがCPUの動作に影響を及ぼすのは、システム（CEC）が区画モードで動作しており、WARが非ゼロ値を含むときだけである。

【0072】ゲスト位置にアクセスするためのプロセス—図4および図5

ホストは、割込みまたはインタセプトの処理中に検出されたゲスト位置へのアドレス指定可能性を獲得するために、図4または図5のハードウェアおよびプロセスを使用する。ゲスト位置は、ホストがゲスト命令をシミュレートするためにしばしばアクセスする。図4では1つのWARを、図5では複数のWARを使用しているが、その他の点では、どちらの場合も同様に動作する。

【0073】インタセプト・コードは、インタセプトされたゲスト命令をホストが検査しなければならないことを示すことができる。ホスト・ソフトウェアは、ゲストのSDエントリにアクセスし、ゲストのゾーン番号（ZN）と、インタセプトされた命令のスモール・アドレスとを、PSW（プログラム状況ワード）フィールドに取り込む。ホストは次に、ウィンドウ番号を指定し、ウィンドウ活動化命令を実行して、ゲスト位置へのアドレス指定可能性を獲得する。ホストは次に、ゲスト・ゾーン内のゲスト命令にアクセスし、ホスト・ゾーン内のホストによって指定される位置にこの命令をコピーするためのホスト命令に、各ゲスト・オペランド用の関連するゲスト・ウィンドウPFを入れる。

【0074】さらに詳細に述べると、ホストは、このアクセスを実行するために、そのウィンドウ番号のうちの1つを選択し、選択したウィンドウ番号をR1汎用レジスタ42に入れ、ゲストのZNとPSWアドレスをR3レジスタ43に入れ、次いでウィンドウ活動化命令を呼び出して、ラージ・アドレスを生成し、選択したウィンドウ番号と関連するWRにそれを格納する。このようにして、ホストは、図6の流れ図に示すプロセスを実行する際に図4または図5に表されたハードウェアを使用して、ゲスト命令にアクセスする。

【0075】図4または図5で、スモール（実）アドレスが、ゲストのSDへのホスト・アクセスによって、入力レジスタ541または621に入力され、対応するラージ絶対アドレスが出力レジスタ551または629に出力される。

【0076】ホスト・ウィンドウがホスト命令によってアクセスされるとき、図4または図5で使用されるアドレス選択プロセスは、複数の種類のラージ・アドレスのうち1種類を出力レジスタ551または629に出力する。ラージ・アドレスの種類には、ゲスト・ゾーン・プレフィクス絶対アドレス、ゲスト・ゾーン逆プレフィクス絶対アドレス、またはプレフィクス・アドレスでも逆プレフィクス・アドレスでもないゲスト・ゾーン絶対アドレスがある。

【0077】ホストが、そのどのウィンドウに対するものでもない記憶域オペランドを指定するとき、そのオペランドはホスト・ウィンドウ内でアクセスされる。すなわち、ホスト命令が、WARによってホスト・ウィンドウと定義されていないホストPFにアクセスするとき、図4または図5で使用されるアドレス選択プロセスは、出力レジスタ551または629にホスト・ゾーン・ラージ絶対アドレスを出力する。

【0078】入力されたスモール・アドレスが実アドレス・ゼロである場合、図4または図5においてプレフィクシング・プロセスが動作して、それぞれのCPUのプレフィクス・レジスタ中に提供された所定の非ゼロ絶対アドレスを実アドレス0で置き換える。システムの各CPUは、そのプレフィクス・レジスタ（図4または図5のレジスタ544）中に事前に割り当てられた異なるプレフィクス・ラージ・アドレスを持っているので、（割込み値など、そのCPUだけに値を含む）異なるプレフィクス・ページ・フレームにアクセスする。プレフィクス変換動作で、そのプレフィクス・レジスタ中のプレフィクス・アドレスをスモール実アドレス0で置き換えることによって、絶対アドレスが生成される。また、図4または図5で、ラージ・アドレス・ゼロをCPUのプレフィクス値に等しい任意のスモール実アドレスで置き換える逆プレフィクシング・プロセスが実行されるので、逆プレフィクス動作では必ずすべてのCPUについてページ・フレーム0にアクセスする。

【0079】図4および図5のそれぞれで、入力レジスタ541または621中に入力されたホスト・スモール実アドレスのビット1ないし19が、（プレフィクシングのために）比較機構542でゼロと比較され、同時に（逆プレフィクシングのために）比較機構543で、CPUのプレフィクス・レジスタ544中のビット1ないし19によって提供されるプレフィクス値と比較される。比較機構543または542がcompare equal条件のとき、プレフィクス・レジスタ544中のラージ・アドレスまたは逆プレフィクス・ゼロ・ラージ・アドレスは、それぞれ選択機構550または628に送られ、選択機構550または628は、それぞれのラージ・アドレスを出力レジスタ551または629に出力する。

【0080】図4と図5は、ホスト・ウィンドウPFが選択される方法が異なる。図4と図5のどちらでも、ホスト・オペランド・スモール・アドレスの値が、WRの選択を制御する。図4では、ホスト・ウィンドウPFの選択が、そのホスト入力レジスタ541中のビット19の状態によって決定される。すなわち、入力レジスタのビット19の状態が0のときは、WR0のウィンドウが選択され、ビット19の状態が1のときは、WR1のウィンドウが選択される。図5では、各WRがそれぞれ異なるWARと関連付けられ、（それに含まれるホスト・ウィンドウPFのビット1ないし19が入力ビット1な

いし19に等しいWARと関連付けられた)WRが選択される。

【0081】図4でさらに詳細に説明すると、スモール・ホスト実アドレスをラージ絶対アドレスに変換する手順は、次のプロセスによって示される。

【0082】1. 入力アドレス・ビット1ないし18がWAR 545中のビット1ないし18に等しく、入力レジスタのビット19が0であり、WR0の内容がゼロに等しくないときは、WR0で指定される絶対ページ・フレーム・アドレスで、入力アドレスのビット1ないし19が置き換えられる(ラージ絶対アドレス・ビット33ないし51を置き換える)。WR0中の最上位ビット1ないし32は影響を受けない。これらのビットは、図3で提供されるゾーン原点ビットのままである。WR0中の最下位ビット52ないし63も影響を受けない。これらのビットは、入力されたスモール・アドレスからの変位ビット20ないし31である。

【0083】2. 入力アドレス・ビット1ないし18がWARビット1ないし18に等しく、入力レジスタ・ビット19が1であり、WR1の内容がゼロに等しくないときは、WR1で指定される絶対ページ・フレーム・アドレスがアドレスのビット1ないし19と置き換わる。

【0084】3. 入力された実アドレス・ビット1ないし18がWARビット1ないし18に等しくないとき、あるいは入力されたビット19が0であり、WR0の内容がゼロに等しい場合、あるいは入力されたビット19が1であり、WR1の内容がゼロに等しい場合、アドレスは元のままである。

【0085】WARの内容がゼロに等しいとき、出力されるラージ・アドレスにおいて、入力されたアドレスは元のままである。なぜなら、ホスト・ウィンドウが選択されていないからである(ホスト・ウィンドウ0はウィンドウとして使用できない)。この場合、出力されるアドレスは、入力されたアドレスにおいて、ホスト・ゾーン原点がスモール・アドレスの最上位端と連結されたものである。

【0086】いずれの場合も、入力レジスタ541のスモール・アドレスのDビット20ないし31は、変更されずに、出力されるラージ・アドレスに移される。

【0087】WARのビット1ないし19がホスト・プレフィクス・レジスタのビット1ないし19に等しいときは、プレフィクス・レジスタ544中のスモール絶対アドレスがラージ絶対アドレスとして出力される。

【0088】図4では、選択機構548がスモール実アドレス541の最下位ビット19を使用して、2つのウィンドウ・レジスタ546および547の一方を選択する。WARによって3つ以上のホスト・ウィンドウPFが提供される場合、選択機構548は、入力されたスモール・アドレスのビット19を通じて複数の最下位ビットを受け取り、それぞれがウィンドウPFと関連するn

個のWRのうちの1つを選択する。

【0089】図5では、選択機構627は図4の選択機構548と同様に動作する。しかし、図5では、選択機構627は入力されたスモール・アドレス・ビット1ないし19を各WARのビット1ないし19と比較し、等しいビット1ないし19を持つWARのWRを選択する。

【0090】図4では、別の選択機構550が、実アドレス541、プレフィクス・レジスタ544、選択機構548で選択されたウィンドウ・レジスタ、およびゼロの値のうちの1つを次のように選択する。入力レジスタ541中の入力された実アドレスがゼロに等しい(とボックス542で判定された)場合、プレフィクス・レジスタ544が選択される。実アドレス541がプレフィクス・レジスタ544に等しい(とボックス543で判定された)場合、ゼロの値が選択される。入力レジスタ541中の入力されたビット1ないし19がWAR 45のビット1ないし19に等しい(とボックス549で判定された)場合、選択機構548で選択されたWR値が出力される。これらの比較がいずれも等しくならない場合、レジスタ541中の入力されたアドレス・ビット1ないし19が、ホスト・ゾーンにアクセスするため、ラージ・アドレス・ビット33ないし51として出力選択機構に直接入れられる。

【0091】図5では、選択機構628は図4の選択機構550と基本的に同じ動作をする。

【0092】選択機構550または628中のラージ絶対アドレスの値は最終的に、出力レジスタ551または629に移され、このレジスタから、アドレスを使用して大型メモリがアクセスされる。

【0093】ホストによるゲスト位置へのアクセス—図6

図6は、図4または図5に示すハードウェアがラージ絶対アドレスを生成するのに使用するプロセスを表している。各図で、入力されたスモール実アドレスがラージ絶対アドレスに変換される。アドレス変換プロセス中、処理のプレフィクス/ゼロ部分はWR/WAR選択処理と並行して実行することができる。

【0094】処理は、ステップ771から開始する。ステップ772に進み、入力アドレスのビット1ないし19がゼロの値と比較される。入力ビット1ないし19がゼロに等しい場合、ステップ773で、絶対アドレスがプレフィクス・レジスタの内容に設定される。ステップ772で入力ビット1ないし19がゼロに等しくないと判定された場合、ステップ774で、これらの入力ビットが、プレフィクス・レジスタの対応するビットと比較される。ステップ772でこれらの入力ビットどうしが等しい場合、ステップ775で、絶対アドレスがすべてゼロに設定される。ステップ774でこれらのビットが等しくないと判定された場合、ステップ776で、入力

ビット1ないし18がWARビット1ないし18と比較されて、ウィンドウが使用中かどうか判定される。

【0095】ビット1ないし18が等しい場合、ステップ778で、入力されたアドレスの下位ビット19が復号されて、WR0とWR1のどちらを選択するかが決定される。

【0096】次に、ステップ779で、入力アドレス・ビット1ないし19が、絶対アドレス中の適切なウィンドウ・レジスタ・ビット1ないし51と置き換えられて、出力アドレス781が提供される。

【0097】ステップ776で入力されたビット1ないし18が、WAR中の対応するPFビットと等しくないと判定された場合、ステップ777に進み、入力されたアドレス・ビット1ないし19が絶対アドレス・ビット位置33ないし51に格納され、これによって出力される絶対アドレス780が提供される。

【0098】絶対アドレス780または出力アドレス781は、要求されたアクセスを行うために、大型メモリの制御装置に提供される。

【0099】ゾーンの再配置—図7

図7は、1つまたは複数のCPUを持つことが可能なシステム内で大型メモリを再構成するためのCPUゾーン・アドレス再配置のフロー・チャートを表す。ゾーンは、システム内のホスト区画および論理区画の間で再構成される。大型記憶域におけるゲスト・ゾーンまたはホスト・ゾーンの位置の変更は、ホスト・プログラムおよびゲストに対して透過的である。任意の既存のゲスト・ゾーンの原点または限界が変更され、任意のゲスト・ゾーンが削除され、1つまたは複数の新規ゲスト・ゾーンが追加される可能性がある。ゲスト・ゾーン原点が変更される場合、そのゾーンが大型メモリ中で再配置される。この場合、WR0またはWR1中のラージ絶対アドレスのどちらか一方または両方を変更しなければならない。

【0100】ホスト・ゾーンは暗示的（ZIT中で定義されない）でも、明示的（ZIT中で定義される）でもよい。暗示的な場合、ホスト・ゾーンが再配置されることはなく、ホスト限界の検査は行われない。明示的な場合、各WARの内容と、ZITホスト・エントリ（たとえば、ZN0をもつエントリ）中のホスト・ゾーン限界の間で検査が行われる。

【0101】任意のゲスト・ゾーン限界が変更される可能性がある（ホスト・ウィンドウが使用中の）各CPUウィンドウ指定テーブル・エントリ中の各スモール・アドレスを、ゲスト・ゾーン限界と突き合わせて検査する必要がある。ここで、限界を超えている場合、（CPUウィンドウ指定テーブル・エントリ中、および対応するWR中の）スモール・アドレスと、それに対応するラージ絶対アドレスを無効にしなければならない。

【0102】再構成を実行するには、システム内のCP

Uのうちの1つによって特殊コマンドが発行され、ステップ91でゾーン再配置プロセスが開始する。このコマンドは、システム動作中にいつでも発行することができる。なぜなら、図7のプロセスは、任意のゾーンが再配置、追加、または削除された後にシステム動作を続行するのに必要なアドレス調整を扱うからである。

【0103】次にステップ92に進むと、システム管理者（人間）、またはホスト・プログラム、または論理区画中の割り当てられたゲスト・プログラムが、既にプログラム可能メモリ中で置換ZIT（replacement ZIP）をセットアップしている。置換ZITは、システム内の任意のゾーンについて未変更のゾーン原点またはゾーン限界値を含み、任意のゾーン原点値またはその限界値を変更し、1つまたは複数のゾーン・エントリを新規ゾーンとして追加し、あるいは1つまたは複数のゾーン・エントリを削除してシステムからそのゾーンを削除することができる。現在活動状態の（置き換えられる）ZITは、（プログラム・アドレスではアクセスできない）「システム・エリア記憶域」にある。

【0104】次にステップ93で、新規ZITの置換および活動化を可能にするために、特殊コマンドによる動作で、システム内のすべてのCPUによる実行が一時的に中断される。ZITが置き換えられている間に、CPUがこのZITにアクセスできるようにすることはできない。そのようにすると、変更プロセス中にエラーが発生する可能性があるからである。

【0105】新規ZITの活動化はステップ94で行われる。活動化プロセッサは、システム内のCPUの1つ、または特殊プロセッサ（たとえば、サービス・プロセッサまたは何らかの補助プロセッサ）とすることができる。活動化プロセスは、特殊コマンドのハードウェア／マイクロコードにより実行される。すなわち、置換ZITを、メモリ内のプログラム可能領域から、WR命令ハードウェア／マイクロコードが活動状態のZITにアクセスするシステム・エリア記憶域内の位置に存在する古いZIT上にコピーすることによって実行される。

【0106】ステップ95で、特殊コマンドが、新たに活動化されたZITの内容とCPUウィンドウ指定テーブルの内容を使用して、各CPU内のWARおよびWRの内容について必要な検査および再計算を実行する。ZIT中でゾーンが明示的に指定されている場合、WARの内容がどの新規限界をも超えないことを確認するために、各WAR中の値がホスト・ゾーン限界と突き合わせて検査される。新規限界を超えた場合、関連するWARの内容が無効にされ（かつ、そのWARによって定義されるウィンドウと関連するWRも無効にされ）、ホストに、WAR中でアドレス指定されたウィンドウPFの位置を変更するためにWARロード命令を再発行するよう要求する、例外信号が生成される。限界検査によって、ホスト・ゾーン限界を超えていないと判定された場合、

WAR中のスモール・アドレスが有効になる。

【0107】各WARの内容が検査された後、各CPUウィンドウ指定テーブル・エントリ（図8を参照）中で表された各WRの内容を検査し再計算することによって、ステップ95のプロセスが続行する。各CPUウィンドウ指定テーブル・エントリがアクセスされ、それに含まれるゲスト・ゾーン識別子およびゲストのスモール・アドレスが再計算に使用される。ゾーン識別子は、ゲスト・ゾーンのZITエントリにアクセスするのに使用される。その際、限界を超えていないかどうか検査するために、ゾーンの限界がゲストのスモール・アドレスと比較される。限界を超えている場合、ゲスト・スモール・アドレスが無効化され、エントリ中のラージ・アドレスが無効化される（たとえば、ラージ・アドレスがゼロに設定され、あるいは有効ビットがオフに設定される）。ゲストのスモール・アドレスが限界内である場合、図3でウィンドウ・エントリに関して示したウィンドウ活動化命令プロセスを使用することによって、テーブル・エントリ中の対応するラージ絶対アドレスが再計算され、再計算された値が、対応するWRおよびCPUウィンドウ指定テーブル・エントリに入れられる。CPUウィンドウ指定テーブルのいずれかのゾーンが新規ZIT中にない場合、そのゾーンは削除されている。その場合、スモール・アドレス・フィールドとラージ・アドレス・フィールドはどちらも、CPUウィンドウ指定テーブル中のそのウィンドウ・エントリについて無効に設定され、関連するWRエントリの内容が無効に設定される。

【0108】次にステップ96で、システム内のすべてのCPUが、先に中断された場所から続行することによってその通常の命令処理を再開する。次に、特殊命令動作が完了し、ステップ97で終了する。

【0109】CPUウィンドウ指定テーブル—図8

図8は、CPUウィンドウ指定テーブルを表している。（ゲストおよびホストを実行することが可能な）システム内の各CPUは、それ自体のウィンドウ指定テーブルがシステム・エリア記憶域に格納されている。

【0110】CPUウィンドウ指定テーブルは、それぞれのCPUに組み込まれた各ウィンドウ用のそれぞれのエントリを含む。テーブル中のウィンドウ・エントリは、ウィンドウ番号のW0およびW1で索引付けされる。各エントリは、それぞれのウィンドウに現在割り当てられているゾーン番号、関連するWRに現在含まれているラージ絶対アドレス、および同じエントリ中のラージ絶対アドレスによって表されるスモール・アドレスを含む。エントリに現在割り当てられているウィンドウがない場合、エントリにゼロが格納される（あるいは、オプションの有効ビットがオフ状態に設定される）。

【0111】エントリ中の各フィールドは、本明細書の他の節で説明したとおりに処理される。

【図面の簡単な説明】

【図1】本発明の実施例を使用するコンピュータ・システム内にN個のゲスト・ゾーンを有するシステム・メモリとシステム・エリア記憶域の簡略化したマップである。

【図2】ウィンドウ・アクセス・レジスタ（WAR）をロードするための「WARロード」命令によって使用されるハードウェアおよびプロセスを表す図である。

【図3】大型メモリの（任意のゾーンにおける）任意の位置にアクセスすることを目的として、任意のウィンドウ・レジスタ（WR）にラージ・アドレスをロードするために「ウィンドウ活動化」命令によって使用されるハードウェアおよびプロセスを表す図である。

【図4】ゲスト・ラージ・アドレスを出力し、このアドレスでホストのオペランド・スモール・アドレスを置き換えるためにWRを選択する、好ましい実施例で 사용되는プロセスを表す図である。

【図5】ホスト・ウィンドウPFに等しいホスト・オペランド・スモール・アドレスをラージ・アドレスで置き換えるためにWRを選択する別のプロセスを表す図である。

【図6】大型メモリ内の任意の位置にアクセスするためにラージ・アドレスを生成する際にスモール・ホスト・アドレスを使用し、アドレス置換によって任意のゲスト・ゾーン位置にアクセスするプロセスのフロー・チャートである。

【図7】システムのメモリ内のゲスト・ゾーンおよびホスト・ゾーンを再構成するために、大型メイン・メモリ内の各ゾーンを動的に再配置（変更）するプロセスのフロー・チャートである。

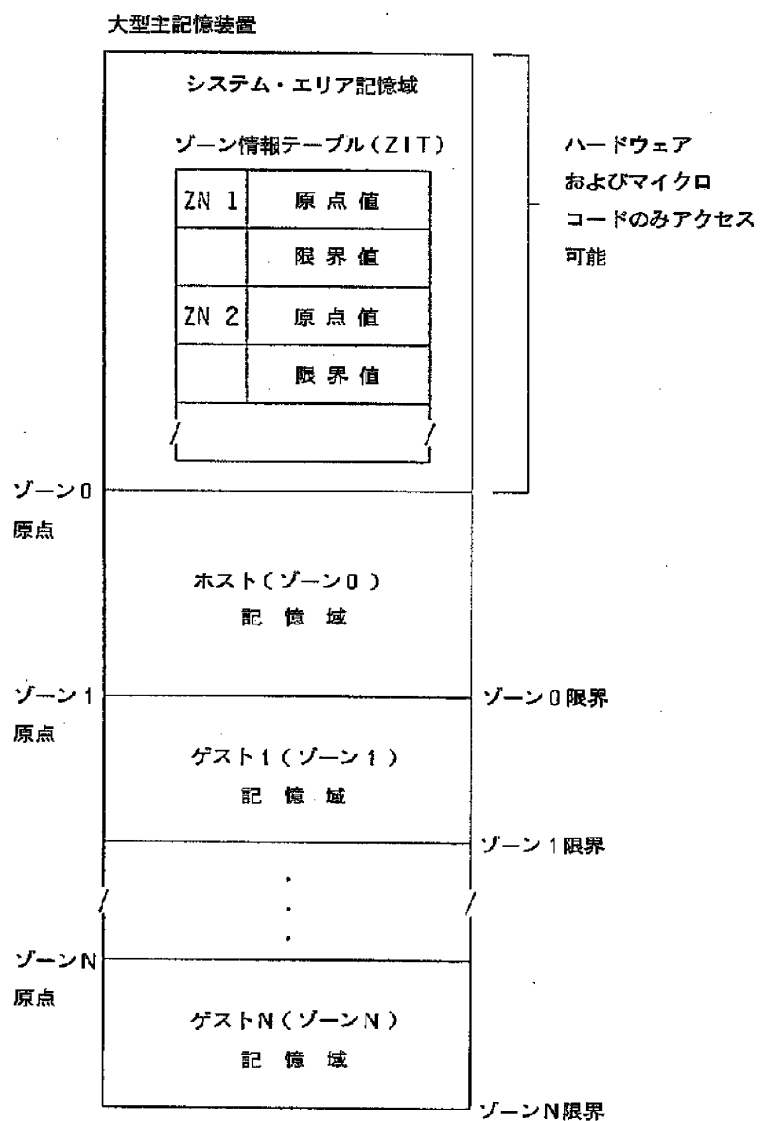
【図8】動的ゾーン再配置プロセスで使用するCPUウィンドウ指定テーブルを表す図である。

【図9】ホストによるアクセスを必要とするゲスト・ゾーン内の位置をホストが決定するためのホスト・テーブルにおける状態記述（SD）エントリの例を示す簡略化した図である。

【符号の説明】

- 42 R1汎用レジスタ
- 43 R3汎用レジスタ
- 44 ゾーン情報テーブル（ZIT）
- 45 比較機構
- 47 CPUウィンドウ指定テーブル・エントリ
- 48 ウィンドウ・レジスタ（WR）
- 49 加算機構
- 541 入力レジスタ
- 542 比較機構
- 544 プレフィクス・レジスタ
- 545 ウィンドウ・アクセス・レジスタ（WAR）
- 550 選択機構
- 551 出力レジスタ

【図1】



【図8】

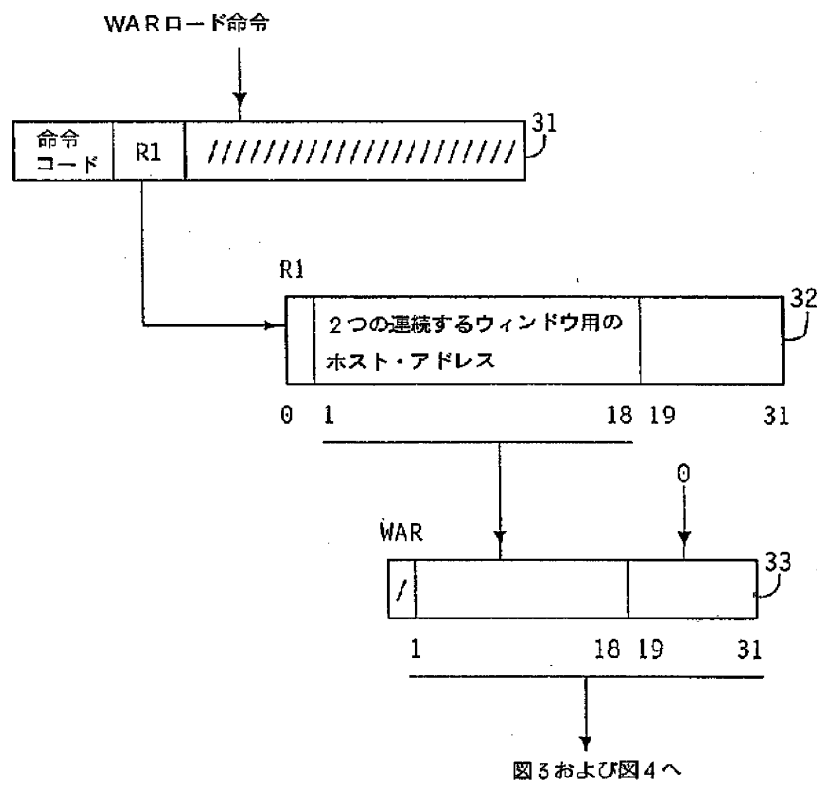
ウィンドウ

番号

CPUウィンドウ指定テーブル

W0	ZN	ゲスト・アドレス (スモール)	ラージ絶対アドレス
W1	ZN	ゲスト・アドレス (スモール)	ラージ絶対アドレス
/ / / /			
Wn	ZN	ゲスト・アドレス (スモール)	ラージ絶対アドレス

【図2】



指定された2つの連続するPFはそれぞれWR0およびWR1と
関連付けられる

【図3】

ウィンドウ活動化命令

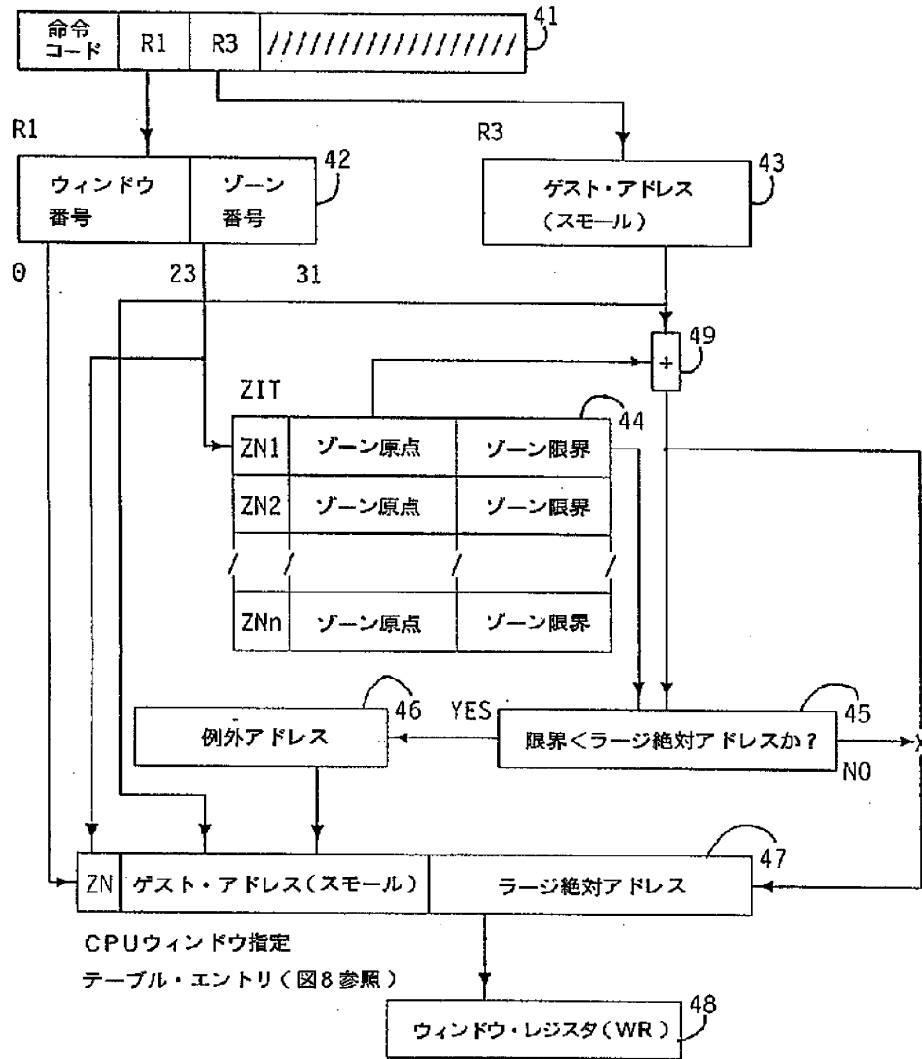
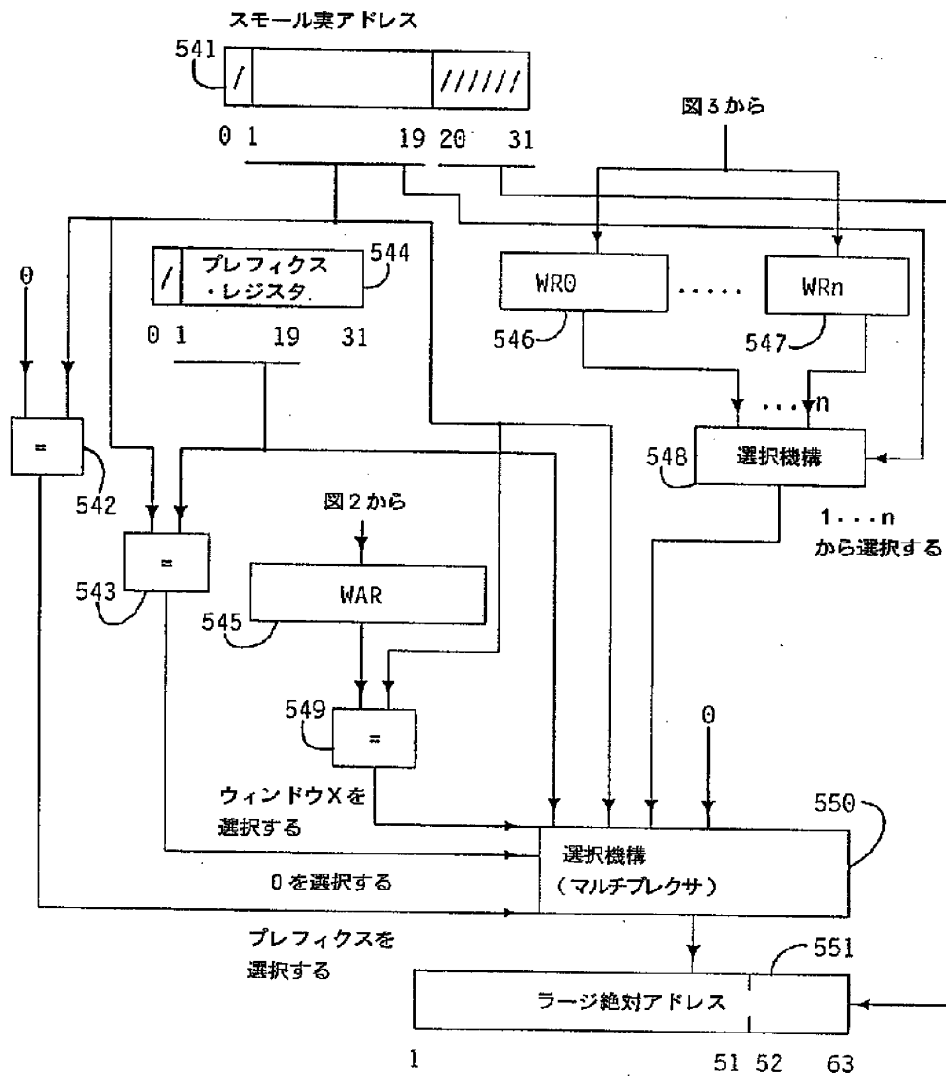
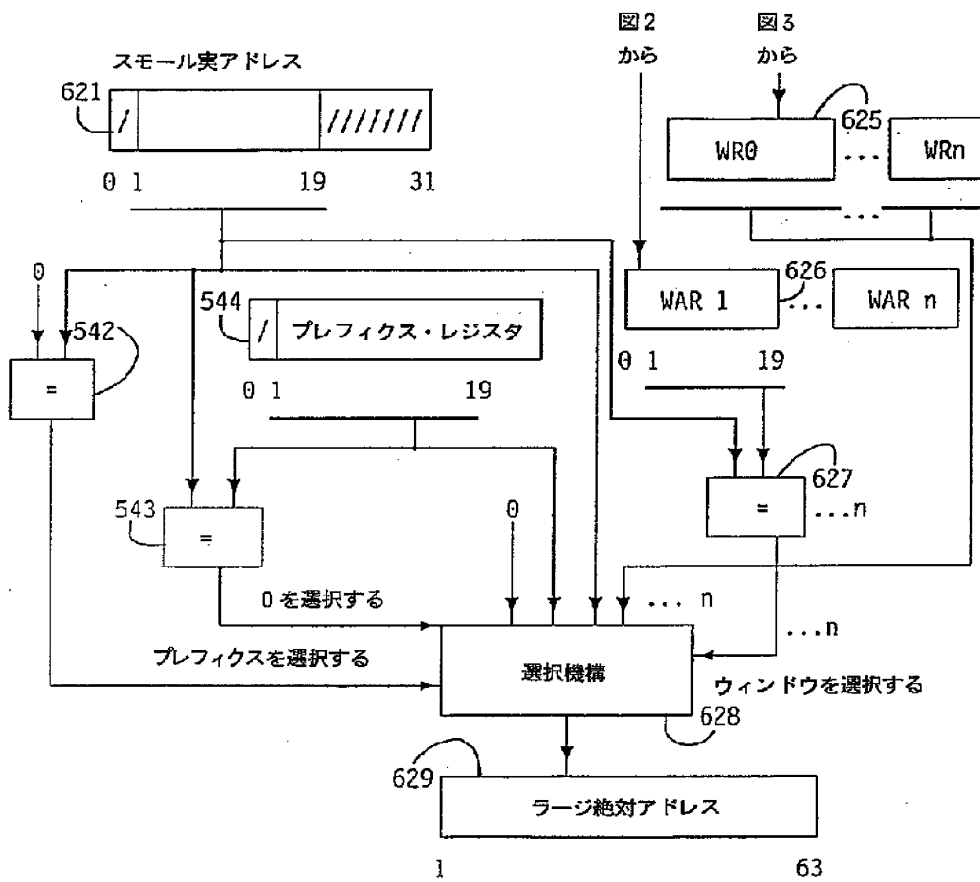


図4および図5へ

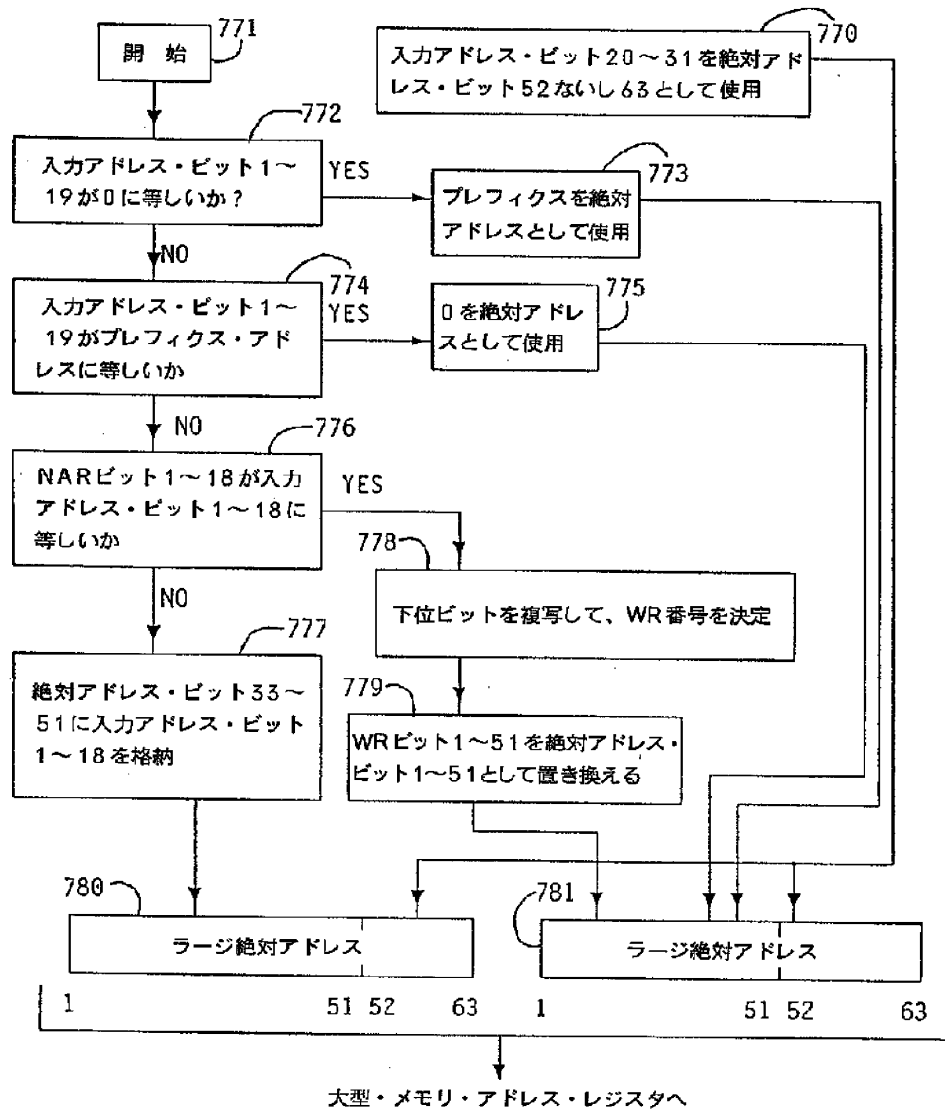
【図4】



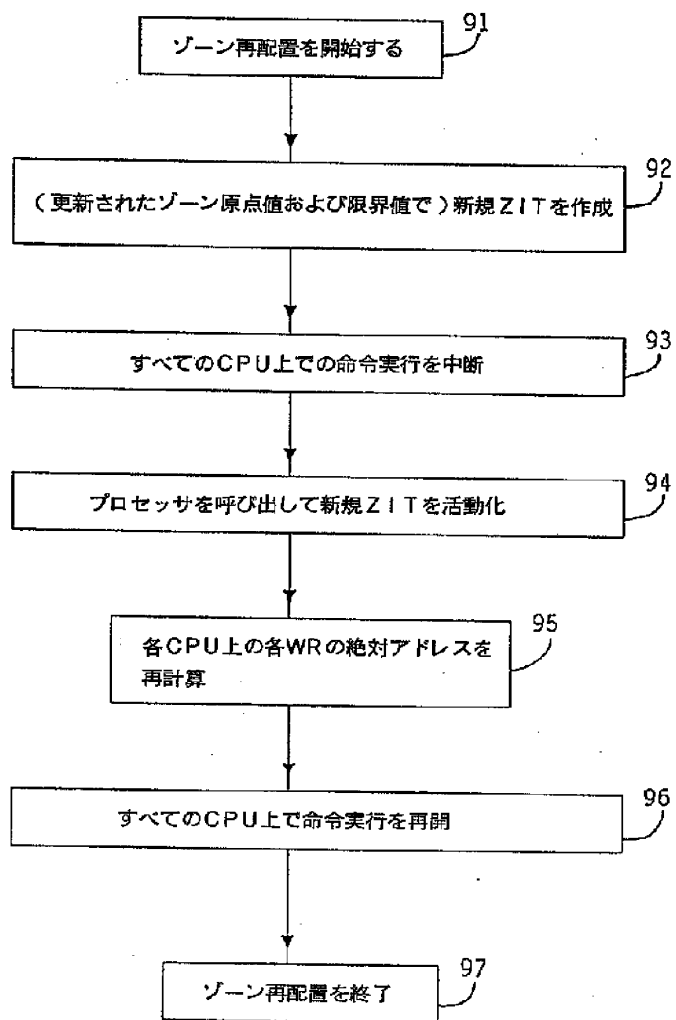
【図5】



【図6】

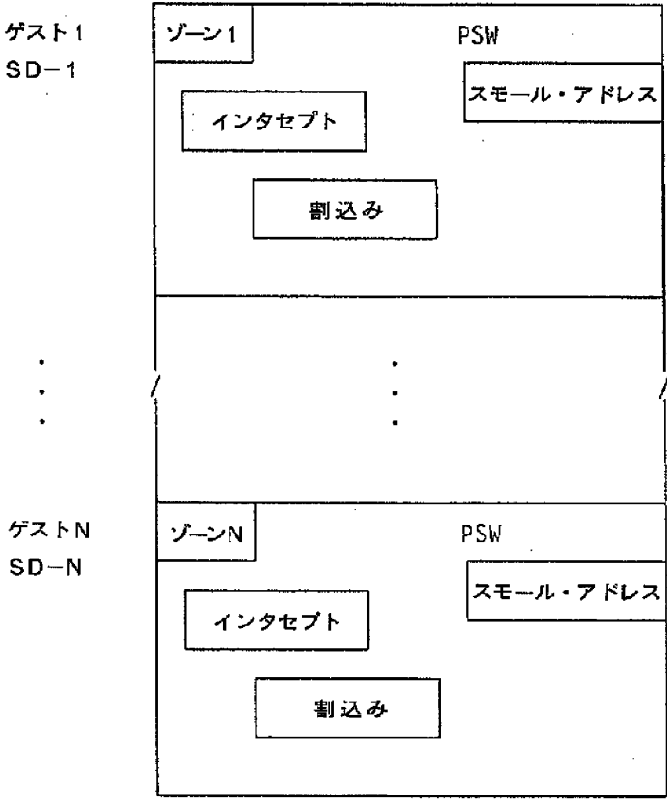


【図7】



【図9】

(ホスト・ゾーン中の)SDテーブル



フロントページの続き

(72)発明者 ロジャー・エルドレッド・フー
アメリカ合衆国12528、ニューヨーク州ハ
イランド、シャロン・ドライブ 7
(72)発明者 モーン・ジュ・キム
アメリカ合衆国12590、ニューヨーク州ワ
ッピングーズ・フォールス、シャーウッ
ド・ロード 26

(72)発明者 アレン・ヘルマン・プレストン
アメリカ合衆国12601、ニューヨーク州ボ
ーキープシー、イースト・シー・ストリー
ト 97
(72)発明者 デーヴィッド・エメット・ストウッキ
アメリカ合衆国12603、ニューヨーク州ボ
ーキープシー、フォックス・ラン 123
(72)発明者 チャールズ・フランクリン・ウェッブ
アメリカ合衆国12603、ニューヨーク州ボ
ーキープシー、マイネッティ・ドライブ
4